

# **Typical test task for SDET**

# Table of Contents

0. Preconditions.....	3
0.1 Create a new GitHub repository.....	3
0.2 Clone the new repository to your PC.....	3
0.3 Initialize the new npm project.....	3
0.4 Install cypress 9.6.0.....	3
0.5 Change the default run script that opens cypress GUI.....	3
0.6 Change the default cypress settings.....	3
0.7 Change the default cypress settings in cypress/support/index.js.....	3
1. Project architecture.....	4
2. Home Page.....	5
3. Login Page.....	6
4. Register Page.....	9
5. Run Scripts.....	12

# 0. Preconditions

## 0.1 Create a new GitHub repository

## 0.2 Clone the new repository to your PC

## 0.3 Initialize the new npm project

CLI=> `npm init -y`

## 0.4 Install cypress 9.6.0

CLI=> `npm i cypress@9.6.0`

## 0.5 Change the default run script that opens cypress GUI

In file package.json

```
"scripts": {  
  "test": "npx cypress open"  
},
```

After that action you can start cypress GUI just typing **`npm test`** in the terminal

## 0.6 Change the default cypress settings

In file cypress.json add base url, skip video recording, increase the timeout, and ignore examples specs

```
{  
  "baseUrl": "https://www.shoplth.com/",  
  "video": false,  
  "defaultCommandTimeout": 7000,  
  "ignoreTestFiles": ["**/1-getting-started/**", "**/2-advanced-examples/**"]  
}
```

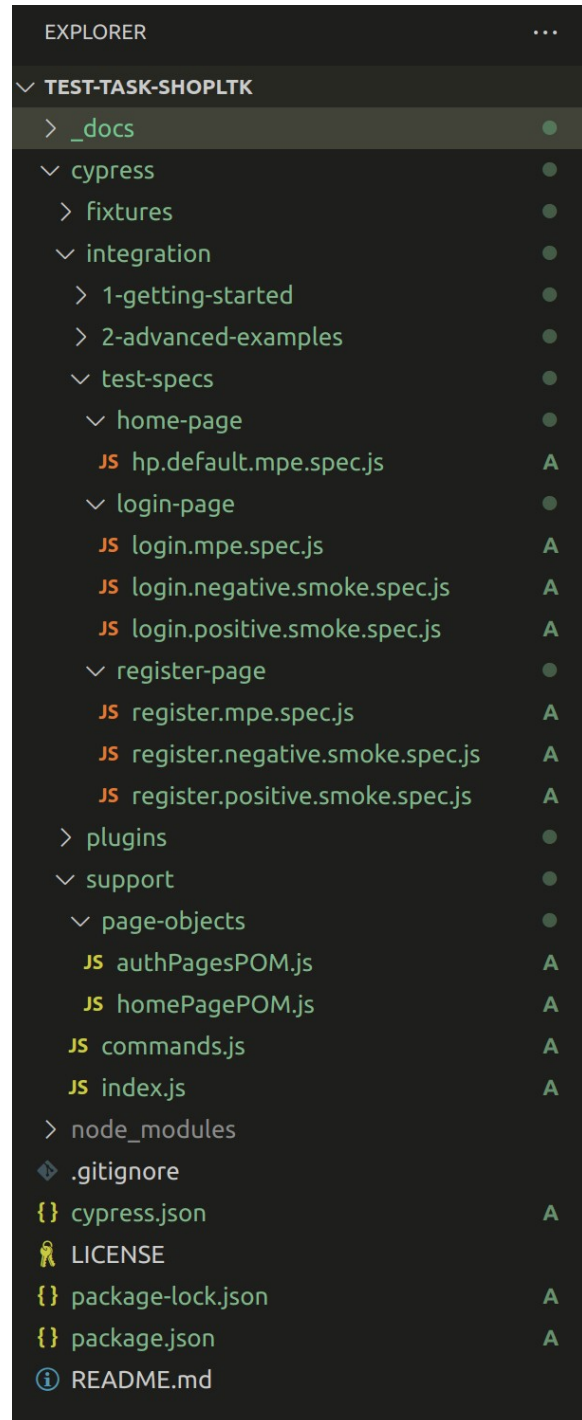
## 0.7 Change the default cypress settings in cypress/support/index.js

```
afterEach() => {  
  //Code to Handle the Sesssions in cypress.  
  //Keep the Session alive when you jump to another test  
  let str = [];  
  cy.getCookies().then((cook) => {  
    cy.log(cook);  
    for (let l = 0; l < cook.length; l++) {  
      if (cook.length > 0 && l == 0) {  
        str[l] = cook[l].name;  
        Cypress.Cookies.preserveOnce(str[l]);  
      } else if (cook.length > 1 && l > 1) {  
        str[l] = cook[l].name;  
        Cypress.Cookies.preserveOnce(str[l]);  
      }  
    }  
  });  
};
```

# 1. Project architecture

Create a new folder **tets-specs** (inside `./cypress/integration/`) - to store all the specs for the project.  
Inside this folder I'm going to create a separate folder for each main page (home, register, login and so on).  
Inside each these folders I'm going to create separate files for each type of tests: main page elements and their base functionality, smoke tests (positive and negative), detailed tests that check each requirement.

Create a new folder **page-objects** (inside `./cypress/support/`) - to store all the Page Object Models for the project.



## 2. Home Page

```
JS homePagePOM.js M ●
cypress > support > page-objects > JS homePagePOM.js > 🏠 HomePage > 📄 load
1  class HomePage {
2    //DEFAULT PAGE - before login
3    testData = {
4      homePage: {
5        default: {
6          pageTitle: {
7            selector: 'h1',
8            text: 'Shop thousands of products tried and styled by real people.',
9          },
10         pageText: {
11           text: 'Shop products across fashion, beauty, home, fitness and more. Tried and styled by real people for real life.',
12         },
13         signUpBtn: {
14           selector: 'a[href="/signup"]',
15           text: 'Sign up',
16         },
17         loginBtn: {
18           selector: 'a[href="/login"]',
19           text: 'Log In',
20         },
21       },
22       login: {
23         pageTitle: {
24           selector: 'h2',
25           text: 'Discover',
26         },
27       },
28     },
29   };
30   //SELECTORS
31   homePageDefaultTitle = () =>
32   | cy.get(this.testData.homePage.default.pageTitle.selector);
33   signUpBtn = () => cy.get(this.testData.homePage.default.signUpBtn.selector);
34   loginBtn = () => cy.get(this.testData.homePage.default.loginBtn.selector);
35   //METHODS
36   //Load Home Page (and check that the Home page was loaded: checking Page Title and Page Text)
37   load = () => {
38     cy.visit('/').location('pathname').should('eq', '/');
39     cy.get(this.testData.homePage.default.pageTitle.selector)
40       .should('be.visible')
41       .should('have.text', this.testData.homePage.default.pageTitle.text);
42     cy.get(this.testData.homePage.default.pageTitle.selector)
43       .next()
44       .should('be.visible')
45       .should('have.text', this.testData.homePage.default.pageText.text);
46     cy.get(this.testData.homePage.default.signUpBtn.selector)
47       .first()
48       .should('be.visible')
49       .should('have.text', this.testData.homePage.default.signUpBtn.text);
50     cy.get(this.testData.homePage.default.signUpBtn.selector)
51       .first()
52       .should('be.visible')
53       .should('have.text', this.testData.homePage.default.signUpBtn.text);
54     //////////////////////////////////////
55     //AFTER LOGIN
56     homePageTitle = () => cy.get(this.testData.homePage.login.pageTitle.selector);
57     userMenuButton = () => {
58       cy.get('button[aria-label="menu"]');
59     };
60   }
61   export default new HomePage();
```

```
JS hp.default.mpe.spec.js A X
cypress > integration > test-specs > home-page > JS hp.default.mpe.spec.js > ...
1  import homePagePOM from '../support/page-objects/homePagePOM';
2
3  describe('Home Page Default (before Log In) - Main Page Elements and their base functionality Main Test Case', () => {
4    describe('PRECONDITIONS: Load Home Page', () => {
5      it('Visit Home Page and check that the page was loaded', () => {
6        homePagePOM.load();
7      });
8    });
9  });
10
```

### 3. Login Page

```
JS login.mpe.spec.js A X
cypress > integration > test-specs > login-page > JS login.mpe.spec.js > describe('Login Page - Main Page Elements and their base functionality Main Test Case') callback >
1 import authPagesPOM from '../../support/page-objects/authPagesPOM';
2 import homePagePOM from '../../support/page-objects/homePagePOM';
3 authPagesPOM;
4 describe('Login Page - Main Page Elements and their base functionality Main Test Case', () => {
5   const loginTitleText = authPagesPOM.testData.AuthPages.pageTitle.textLogin;
6   const registerTitleText =
7     authPagesPOM.testData.AuthPages.pageTitle.textRegister;
8   const loginTextText = authPagesPOM.testData.AuthPages.pageText.loginText;
9   const paragraphText = authPagesPOM.testData.AuthPages.paragraph.loginText;
10  > describe('PRECONDITIONS: Load Home Page', () => {
17  });
18  describe('Checking Page Title', () => {
19    it('Page Title should exist and be visible', () => {
20      authPagesPOM.authPagesTitle().should('be.visible');
21    });
22    it('Page Title should have text "${loginTitleText}"', () => {
23      authPagesPOM.authPagesTitle().should('have.text', loginTitleText);
24    });
25  });
26  describe('Checking Page Text', () => {
27    it('Page Text should exist and be visible', () => {
28      authPagesPOM.authPagesText().next().should('be.visible');
29    });
30    it('Page Text should have text "${loginTextText}"', () => {
31      authPagesPOM.authPagesText().should('have.text', loginTextText);
32    });
33  });
34  describe('Checking Paragraph', () => {
35    it('Page Paragraph should exist and be visible', () => {
36      authPagesPOM.authPagesText().next().should('be.visible');
37    });
38    it('Page Text should have text "${paragraphText} ${authPagesPOM.testData.AuthPages.signUpLink.text}"', () => {
39      authPagesPOM
40        .authPagesParagraph()
41        .should(
42          'have.text',
43          paragraphText + ' ' + authPagesPOM.testData.AuthPages.signUpLink.text
44        );
45    });
46  });
47  describe('Checking Sign Up link', () => {
48    it('Sign Up link should exist and be visible', () => {
49      authPagesPOM.signUpLink().should('be.visible');
50    });
51    it('Sign Up link should have text "${authPagesPOM.testData.AuthPages.signUpLink.text}"', () => {
52      authPagesPOM
53        .signUpLink()
54        .should('have.text', authPagesPOM.testData.AuthPages.signUpLink.text);
55    });
56    it('Sign Up link should redirects user to the Register Page', () => {
57      authPagesPOM.signUpLink().click();
58      authPagesPOM
59        .authPagesTitle()
60        .should('have.text', registerTitleText)
61        .should('be.visible')
62        .go('back');
63    });
64  });
65  describe('Checking Input Fields', () => {
66    it('Checking Email Input Field', () => {
67      authPagesPOM.checkInputField(49, 'email');
68    });
69    it('Checking Password Input Field', () => {
70      authPagesPOM.checkInputField(52);
71    });
72  });
73  > describe('Checking Submit Button', () => {
88  });
89  });
90
```



cypress &gt; integration &gt; test-specs &gt; login-page &gt; JS login.negative.smoke.spec.js &gt; describe('Login Page - SMOKE TEST NEGATIVE (attempt to Log In with random valid credentials (

```
1 import authPagesPOM from '../../support/page-objects/authPagesPOM';
2 import homePagePOM from '../../support/page-objects/homePagePOM';
3 authPagesPOM;
4 describe('Login Page - SMOKE TEST NEGATIVE (attempt to Log In with random valid credentials (
5     const emailForRegister = authPagesPOM.generateEmail();
6     const passwordForRegister = authPagesPOM.generatePassword();
7     describe('PRECONDITIONS: Load Home Page', () => {
8         it('Visit Home Page and check that the page was loaded', () => {
9             homePagePOM.load();
10        });
11        it('Click Log In button', () => {
12            homePagePOM.logInBtn().click();
13        });
14    });
15    describe('Log In action', () => {
16        it('Enter a valid email', () => {
17            cy.get(authPagesPOM.testData.AuthPages.emailInputField.fieldSelector)
18                .type(emailForRegister)
19                .should('have.value', emailForRegister);
20        });
21        it('Enter a valid password', () => {
22            cy.get(authPagesPOM.testData.AuthPages.passwordInputField.fieldSelector)
23                .type(passwordForRegister)
24                .should('have.value', passwordForRegister);
25        });
26        it('Click continue button', () => {
27            cy.get(authPagesPOM.testData.AuthPages.submitBtn.selector).click();
28        });
29        it('Checking the error message', () => {
30            cy.get(authPagesPOM.testData.AuthPages.errorMessage.selector)
31                .children('p')
32                .should(
33                    'have.text',
34                    authPagesPOM.testData.AuthPages.errorMessage.logintext
35                );
36        });
37    });
38 });
39
```

cypress &gt; integration &gt; test-specs &gt; login-page &gt; JS login.positive.smoke.spec.js &gt; ...

```
1  import authPagesPOM from '../.../support/page-objects/authPagesPOM';
2  import homePagePOM from '../.../support/page-objects/homePagePOM';
3  authPagesPOM;
4  describe('Login Page - SMOKE TEST POSITIVE - Main Test Case', () => {
5      const emailForLogin = authPagesPOM.testData.users.test1.email;
6      const passwordForLogin = authPagesPOM.testData.users.test1.password;
7      describe('PRECONDITIONS: Load Home Page', () => {
8          it('Visit Home Page and check that the page was loaded', () => {
9              homePagePOM.load();
10             });
11
12             it('Click Log In button', () => {
13                 homePagePOM.logInBtn().click();
14             });
15         });
16         describe('Log In action', () => {
17             it('Enter a valid email (that was registered before)', () => {
18                 cy.get(authPagesPOM.testData.AuthPages.emailInputField.fieldSelector)
19                     .type(emailForLogin)
20                     .should('have.value', emailForLogin);
21             });
22             it('Enter a valid password (that was registered before)', () => {
23                 cy.get(authPagesPOM.testData.AuthPages.passwordInputField.fieldSelector)
24                     .type(passwordForLogin)
25                     .should('have.value', passwordForLogin);
26             });
27             it('Click continue button', () => {
28                 cy.get(authPagesPOM.testData.AuthPages.submitBtn.selector).click();
29             });
30 >         it('Checking that user was logged in successfully. User profile button should a
38             });
39             it('Log Out', () => {
40                 authPagesPOM.logOut();
41             });
42         });
43     });
44 }
```



## 4. Register Page

```
JS register.mpe.spec.js A X
cypress > integration > test-specs > register-page > JS register.mpe.spec.js > describe('Login Page - Main Page Elements and their base functionality Main Test Case') callback >
1 import authPagesPOM from '.././../support/page-objects/authPagesPOM';
2 import homePagePOM from '.././../support/page-objects/homePagePOM';
3 authPagesPOM;
4 describe('Login Page - Main Page Elements and their base functionality Main Test Case', () => {
5   const registerTitleText =
6     authPagesPOM.testData.AuthPages.pageTitle.textRegister;
7   const loginTitleText = authPagesPOM.testData.AuthPages.pageTitle.textLogin;
8   const registerTextText =
9     authPagesPOM.testData.AuthPages.pageText.registerText;
10  const paragraphText = authPagesPOM.testData.AuthPages.paragraph.registerText;
11  describe('PRECONDITIONS: Load Home Page', () => {
12    it('Visit Home Page and check that the page was loaded', () => {
13      homePagePOM.load();
14    });
15    it('Click Log In button', () => {
16      homePagePOM.signUpBtn().click();
17    });
18  });
19  describe('Checking Page Title', () => {
20    it('Page Title should exist and be visible', () => {
21      authPagesPOM.authPagesTitle().should('be.visible');
22    });
23    it('Page Title should have text "${registerTitleText}"', () => {
24      authPagesPOM.authPagesTitle().should('have.text', registerTitleText);
25    });
26  });
27  describe('Checking Page Text', () => {
28    it('Page Text should exist and be visible', () => {
29      authPagesPOM.authPagesText().next().should('be.visible');
30    });
31    it('Page Text should have text "${registerTextText}"', () => {
32      authPagesPOM.authPagesText().should('have.text', registerTextText);
33    });
34  });
35  describe('Checking Paragraph', () => {
36    it('Page Paragraph should exist and be visible', () => {
37      authPagesPOM.authPagesText().next().should('be.visible');
38    });
39    it('Page Text should have text "${paragraphText} ${authPagesPOM.testData.AuthPages.logInLink.text}"', () => {
40      authPagesPOM
41        .authPagesParagraph()
42        .should(
43          'have.text',
44          paragraphText + ' ' + authPagesPOM.testData.AuthPages.logInLink.text
45        );
46    });
47  });
48  describe('Checking Log In link', () => {
49    it('Log In link should exist and be visible', () => {
50      authPagesPOM.logInLink().should('be.visible');
51    });
52    it('Log In link should have text "${authPagesPOM.testData.AuthPages.logInLink.text}"', () => {
53      authPagesPOM
54        .logInLink()
55        .should('have.text', authPagesPOM.testData.AuthPages.logInLink.text);
56    });
57    it('Log In link should redirects user to the Register Page', () => {
58      authPagesPOM.logInLink().click();
59      authPagesPOM
60        .authPagesTitle()
61        .should('have.text', loginTitleText)
62        .should('be.visible')
63        .go('back');
64    });
65  });
66  describe('Checking Input Fields', () => {
67    it('Checking Email Input Field', () => {
68      authPagesPOM.checkInputField(49, 'email');
69    });
70    it('Checking Password Input Field', () => {
71      authPagesPOM.checkInputField(55);
72    });
73  });
74  describe('Checking Submit Button', () => {
75    it('Checking that Submit Button exists', () => {
76      authPagesPOM.submitButton();
77    });
78    it('Checking that Submit Button has text "${authPagesPOM.testData.AuthPages.submitBtn.registerText}"', () => {
79      authPagesPOM
80        .submitButton()
81        .should(
82          'have.text',
83          authPagesPOM.testData.AuthPages.submitBtn.registerText
84        );
85    });
86    it('Checking that Submit Button is clickable', () => {
87      authPagesPOM.submitButton().click();
88    });
89  });
90 }
```

```
cypress > integration > test-specs > register-page > JS register.negative.smoke.spec.js > describe('Register Page - SMOKE TEST NEGATIVE (attempt to
1  import authPagesPOM from '.././././support/page-objects/authPagesPOM';
2  import homePagePOM from '.././././support/page-objects/homePagePOM';
3  authPagesPOM;
4  describe('Register Page - SMOKE TEST NEGATIVE (attempt to register a new user with email that was already registered)', () => {
5      const emailForLogin = authPagesPOM.testData.users.test1.email;
6      const passwordForLogin = authPagesPOM.testData.users.test1.password;
7      describe('PRECONDITIONS: Load Home Page', () => {
8          it('Visit Home Page and check that the page was loaded', () => {
9              homePagePOM.load();
10          });
11          it('Log Out', () => {});
12          it('Click Sign Up button', () => {
13              homePagePOM.signUpBtn().click();
14          });
15      });
16      describe('Sign Up action', () => {
17          it('Enter a valid email (that was registered before)', () => {
18              cy.get(authPagesPOM.testData.AuthPages.emailInputField.fieldSelector)
19                  .type(emailForLogin)
20                  .should('have.value', emailForLogin);
21          });
22          it('Enter any valid password', () => {
23              authPagesPOM
24                  .getInputById(20)
25                  .type(passwordForLogin)
26                  .should('have.value', passwordForLogin);
27          });
28          it('Click continue button', () => {
29              cy.get(authPagesPOM.testData.AuthPages.submitBtn.selector).click();
30          });
31          it('Checking the error message', () => {
32              cy.get(authPagesPOM.testData.AuthPages.errorMessage.selector)
33                  .children('p')
34                  .should(
35                      'have.text',
36                      authPagesPOM.testData.AuthPages.errorMessage.registerText
37                  );
38          });
39      });
40  });
41
```

cypress &gt; integration &gt; test-specs &gt; register-page &gt; JS register.positive.smoke.spec.js &gt; describe('Register Page - SMOKE TEST POSITIVE - Main Te

```
1 import authPagesPOM from '.././../support/page-objects/authPagesPOM';
2 import homePagePOM from '.././../support/page-objects/homePagePOM';
3 authPagesPOM;
4 describe('Register Page - SMOKE TEST POSITIVE - Main Test Case', () => {
5   const emailRegistration = authPagesPOM.generateEmail();
6   const passwordForRegistration = authPagesPOM.generatePassword();
7   describe('PRECONDITIONS: Load Home Page', () => {
8     it('Visit Home Page and check that the page was loaded', () => {
9       homePagePOM.load();
10    });
11    it('Click Sign Up button', () => {
12      homePagePOM.signUpBtn().click();
13    });
14  });
15  describe('Sign Up action', () => {
16    it('Enter a valid email (that was registered before)', () => {
17      cy.get(authPagesPOM.testData.AuthPages.emailInputField.fieldSelector)
18        .type(emailRegistration)
19        .should('have.value', emailRegistration);
20    });
21    it('Enter any valid password', () => {
22      authPagesPOM
23        .getInputById(20)
24        .type(passwordForRegistration)
25        .should('have.value', passwordForRegistration);
26    });
27    it('Click continue button', () => {
28      cy.get(authPagesPOM.testData.AuthPages.submitBtn.selector).click();
29    });
30    > it('Checking that user was registered successfully. And redirected to the Home Page', () => { ...
38    });
39    it('Log Out', () => {
40      authPagesPOM.logout();
41    });
42  });
43 });
44
```

## 5. Run Scripts

```
{ } package.json > ...
1  {
2    "name": "test-task-shopltk",
3    "version": "1.0.0",
4    "description": "Example of test task",
5    "main": "index.js",
6    "scripts": {
7      "test": "npx cypress open",
8      "mpe": "npx cypress run --spec 'cypress/integration/test-specs/**/*.mpe.spec.js'",
9      "smokePositive": "npx cypress run --spec 'cypress/integration/test-specs/**/*.positive.smoke.spec.js'",
10     "smokeNegative": "npx cypress run --spec 'cypress/integration/test-specs/**/*.negative.smoke.spec.js'",
11     "regression": "npm run mpe && npm run smokePositive && npm run smokeNegative"
12   },
13
14 },
15 "repository": {
16   "type": "git",
17   "url": "git+https://github.com/ILopatenko/test-task-shopltk.git"
18 },
19 "keywords": [],
20 "author": "",
21 "license": "ISC",
22 "bugs": {
23   "url": "https://github.com/ILopatenko/test-task-shopltk/issues"
24 },
25 "homepage": "https://github.com/ILopatenko/test-task-shopltk#readme",
26 "dependencies": {
27   "cypress": "^9.6.0"
28 }
29 }
30
```